

计算机学院专业必修课

计算机组成

时序电路（寄存器）

高小鹏

北京航空航天大学计算机学院
系统结构研究所

Great Idea #1: Levels of Representation/Interpretation

Higher-Level Language
Program (e.g. C)

Compiler

Assembly Language
Program (e.g. MIPS)

Assembler

Machine Language
Program (MIPS)

*Machine
Interpretation*

Hardware Architecture Description
(e.g. block diagrams)

*Architecture
Implementation*

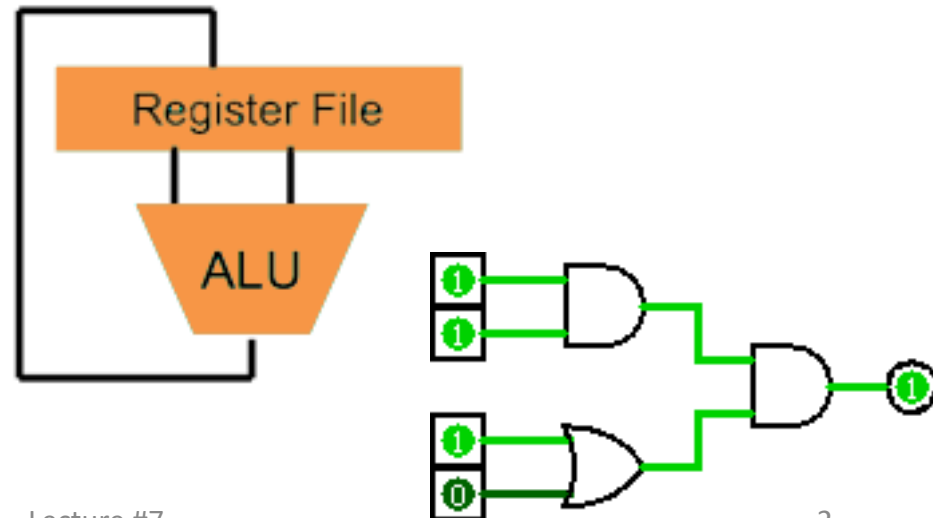
Logic Circuit Description
(Circuit Schematic Diagrams)

```
temp = v[k];  
v[k] = v[k+1];  
v[k+1] = temp;
```

```
lw    $t0, 0($2)  
lw    $t1, 4($2)  
sw    $t1, 0($2)  
sw    $t0, 4($2)
```

We
are
here

```
0000 1001 1100 0110 1010 1111 0101 1000  
1010 1111 0101 1000 0000 1001 1100 0110  
1100 0110 1010 1111 0101 1000 0000 1001  
0101 1000 0000 1001 1100 0110 1010 1111
```



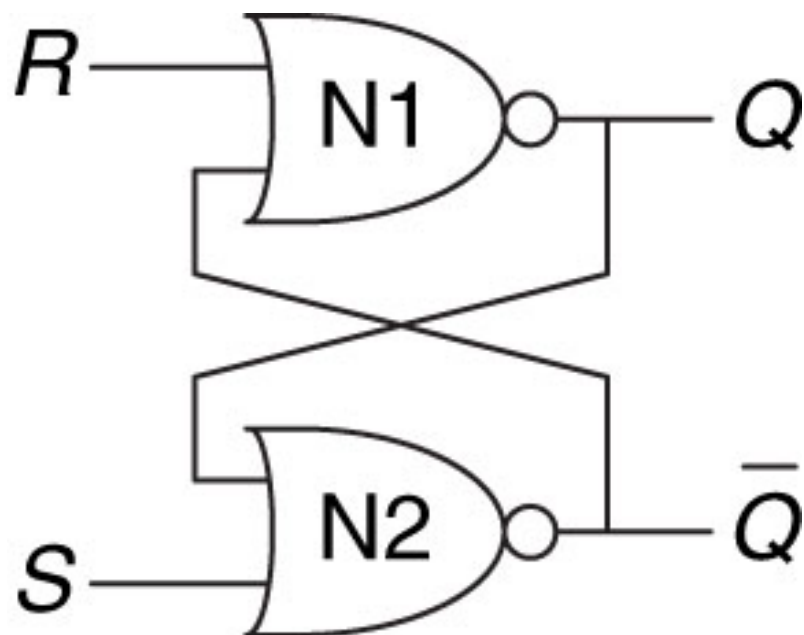
提纲

- 内容主要取材

- CS617的18讲
- 《数字设计和计算机体系结构》的第2章
- 《数字设计和计算机体系结构》的第3章

RS锁存器

- RS锁存器可以自行保持输出状态
 - ▣ 各种触发器的基本构成部分
- 分析RS锁存器的工作原理



R	S	Q	\bar{Q}	
1	0	0	1	
0	1	1	0	
1	1	0	0	非法
0	0	0	1	
		1	0	

RS锁存器

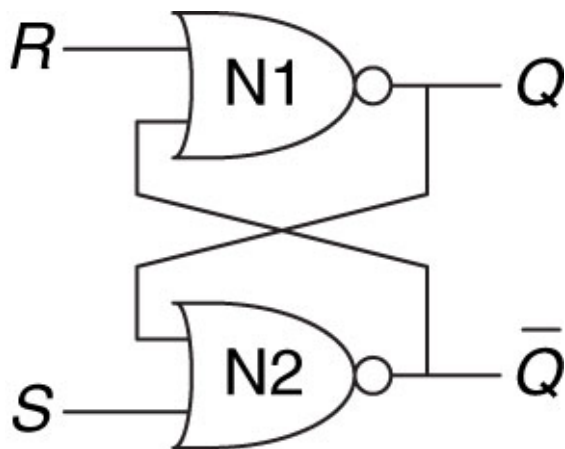
- 结构特征：2个或非门输出信号交叉反馈
 - ◆ 2个输入信号：1个来自外部输入，1个来自另一门的输出
- 输出诉求：2个输出值应该为互补（一个若为0，另一个则1）
- 符号
 - ◆ Q_{prev}/Q ：代表推理前/后的值

变种表示

Q 、 Q^n

Q^n 、 Q^{n+1}

Q 、 Q_{next}

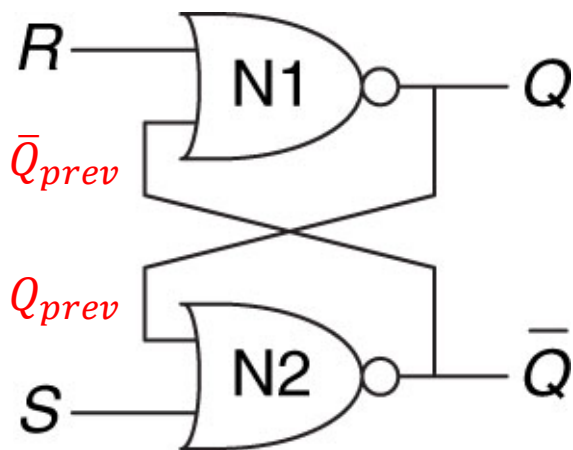


$$Q = \overline{R | \bar{Q}_{prev}}$$

$$\bar{Q} = \overline{R | Q_{prev}}$$

RS锁存器

- 先分析：仅由外部输入就能直接决定输出值的组合
 - 以或非门为例：如果任意外部输入为1，则输出必为0
- 再分析：对于外部输入不能直接决定输出值的组合
 - 假设 Q_{prev} 值然后推理，直至推理得出的 Q 和假设的 Q_{prev} 一致

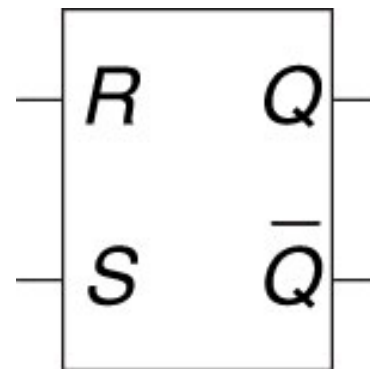


序	S	R	Q_{prev}	Q	\bar{Q}
1	0	1	X	0	1
2	1	0	X	1	0
3	1	1	X	0	0
4.1	0	0	0	0	1
4.2	0	0	1	1	0

使用价值：

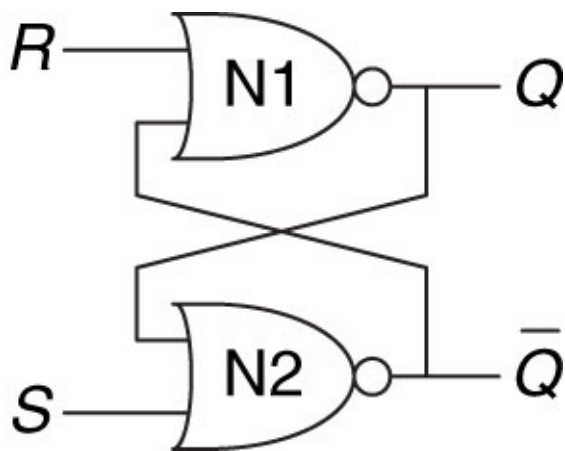
可以先用1或2设置Q的值，
然后用4保持住Q的值

RS锁存器



- 1、去除输出不合理的那组
 - ◆ Q_{prev} 和 Q 同值：不符合规定
- 2、去除 Q_{prev} 输入列

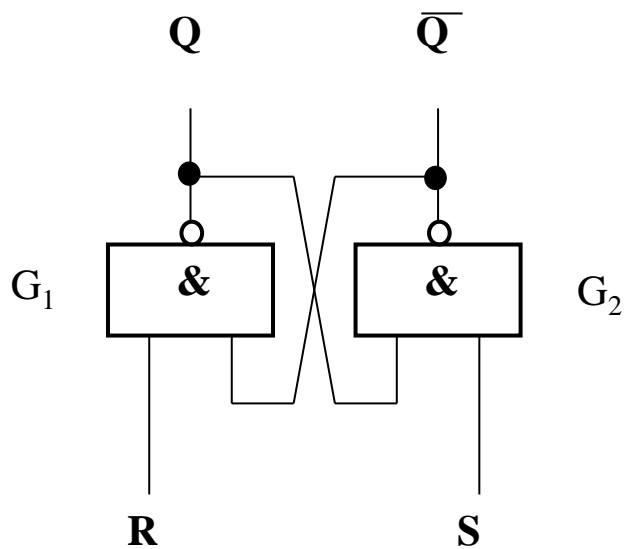
- RS锁存器功能：清除、置位、保持



功能	S	R	Q	\bar{Q}
清除	0	1	0	1
置位	1	0	1	0
保持	0	0	Q_{prev}	\bar{Q}_{prev}
非法	1	1	1	1

用与非门构造锁存器

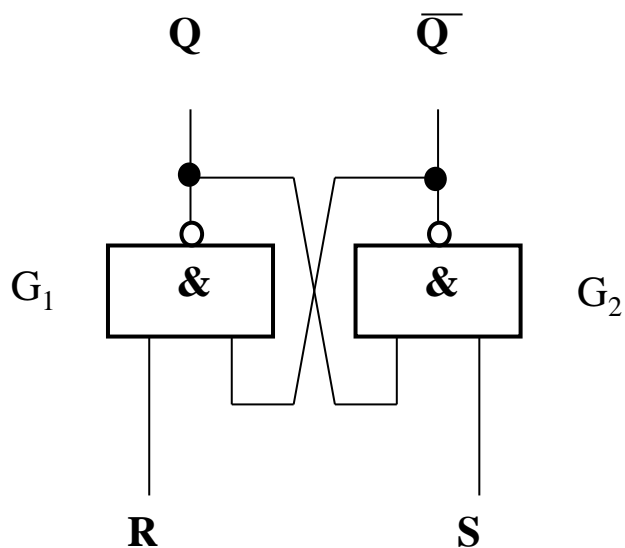
■ 课堂练习



功能	S	R	Q	\bar{Q}
	1	1		
	0	1		
	1	0		
	0	0		

用与非门构造锁存器

■ 课堂练习



功能	S	R	Q	\bar{Q}
保持	1	1	Q_{prev}	\bar{Q}_{prev}
清除	0	1	0	1
置位	1	0	1	0
非法	0	0		

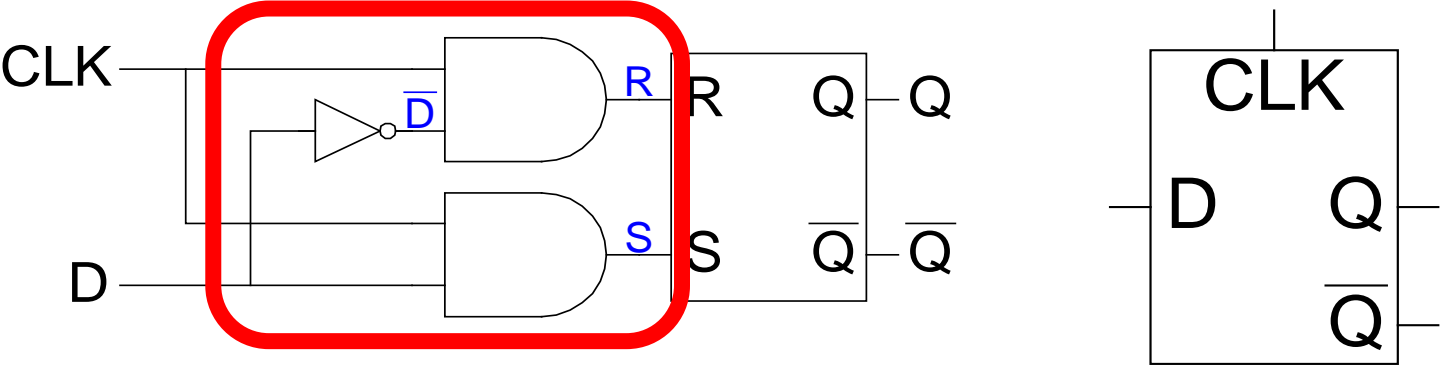
RS锁存器的局限性

- 在数字系统中，为协调各部分电路运行，要求电路在**时钟信号**控制下统一动作
 - 仅在时钟**边沿处**（ $0 \rightarrow 1$ ）保存输入
- RS锁存器本质：没有时间上的同步关系
 - 信号之间难以定序
 - R/S任一改变都可能造成Q改变
 - ◆ R/S的有效值、有效的先后顺序、时间长短均产生影响
- 结果：由于RS锁存器的值与时间难以分离，致使RS难以使用

RS锁存器的局限性

- 解决思路：RS锁存器增加一个控制端
 - 只有控制端有效(1)时，RS锁存器才能动作
 - 即只有控制端有效(1)时，RS锁存器输出由输入决定，否则RS锁存器保持前次值
- 这种控制信号通常为周期振荡信号：方波
 - 也称为时钟信号
 - CLK、Clock

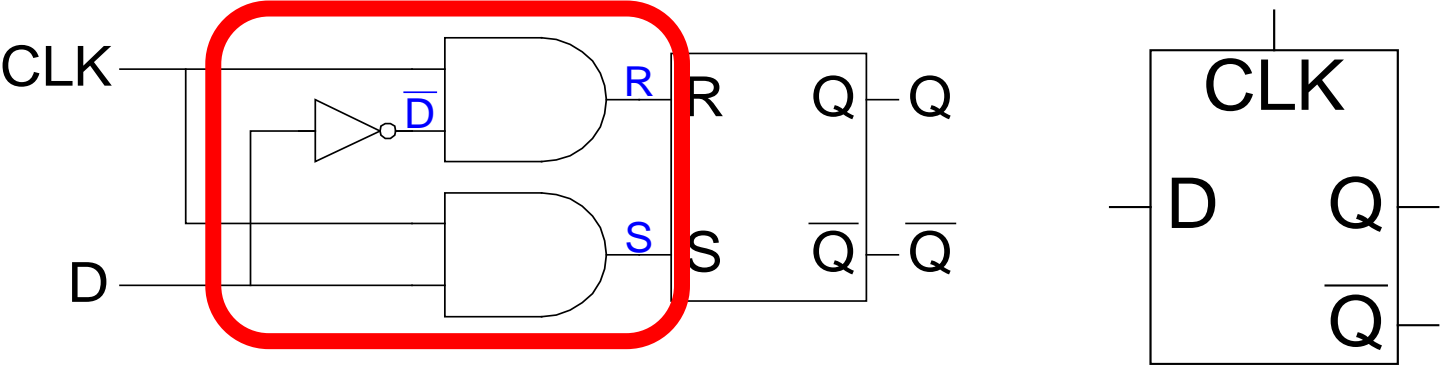
D Latch Internal Circuit^{1/5}



CLK	D	\overline{D}	S	R	Q	\overline{Q}
0	X					
1	0					
1	1					

功能	S	R	Q	\overline{Q}
保持	0	0	Q_{prev}	\overline{Q}_{prev}
清除	0	1	0	1
置位	1	0	1	0
非法	1	1		

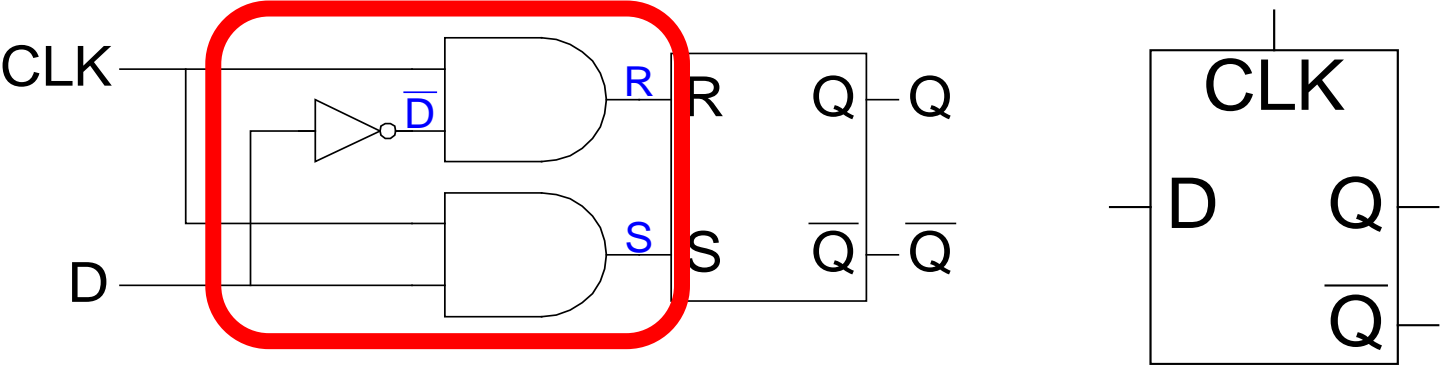
D Latch Internal Circuit^{2/5}



CLK	D	\bar{D}	S	R	Q	\bar{Q}
0	X	X	0	0	Q_{prev}	\bar{Q}_{prev}
1						
1						

功能	S	R	Q	\bar{Q}
保持	0	0	Q_{prev}	\bar{Q}_{prev}
清除	0	1	0	1
置位	1	0	1	0
非法	1	1		

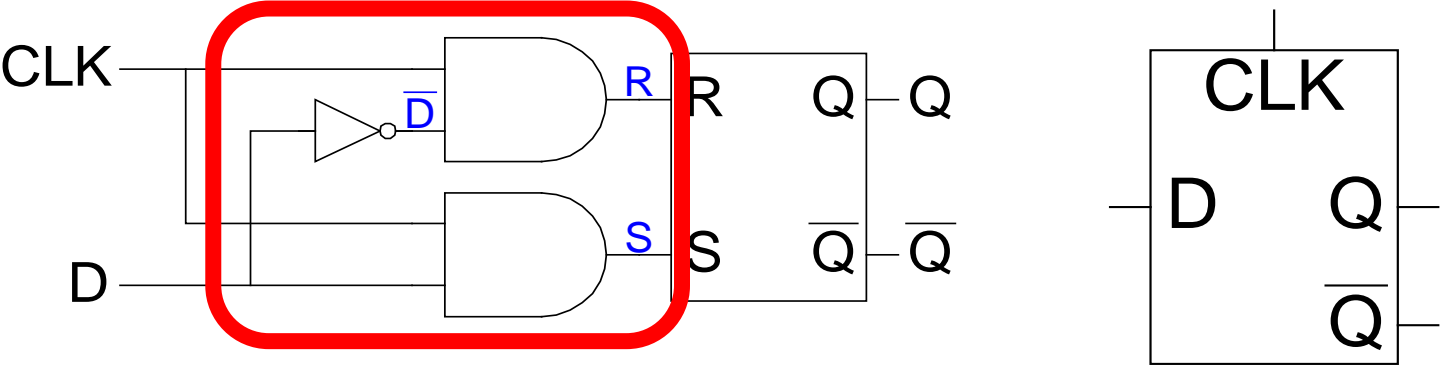
D Latch Internal Circuit^{3/5}



CLK	D	\overline{D}	S	R	Q	\overline{Q}
0	X	X	0	0	Q_{prev}	$\overline{Q}_{\text{prev}}$
1	0	1	0	1	0	1
1						

功能	S	R	Q	\overline{Q}
保持	0	0	Q_{prev}	$\overline{Q}_{\text{prev}}$
清除	0	1	0	1
置位	1	0	1	0
非法	1	1		

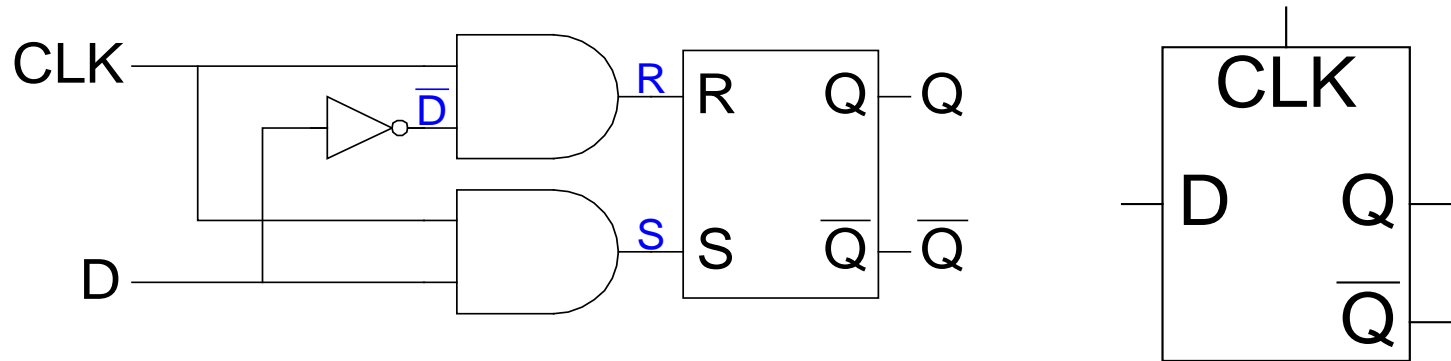
D Latch Internal Circuit^{4/5}



CLK	D	\overline{D}	S	R	Q	\overline{Q}
0	X	X	0	0	Q_{prev}	\overline{Q}_{prev}
1	0	1	0	1	0	1
1	1	1	1	0	1	0

功能	S	R	Q	\overline{Q}
保持	0	0	Q_{prev}	\overline{Q}_{prev}
清除	0	1	0	1
置位	1	0	1	0
非法	1	1		

D Latch Internal Circuit^{5/5}



CLK	D	\bar{D}	S	R	Q	\bar{Q}
0	X	X	0	0	Q_{prev}	\bar{Q}_{prev}
1	0	1	0	1	0	1
1	1	1	1	0	1	0

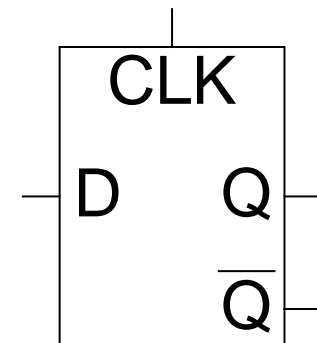
功能	S	R	Q	\bar{Q}
保持	0	0	Q_{prev}	\bar{Q}_{prev}
	0	1	0	1
	1	0	1	0
非法	1	1		

用D和CLK区分开了值与时间

D Latch

- Two inputs: CLK , D
 - CLK : controls *when* the output changes
 - D (the data input): controls *what* the output changes to
- Function
 - When $CLK = 1$,
 D passes through to Q (*transparent*)
 - When $CLK = 0$,
 Q holds its previous value (*opaque*)
- Avoids invalid case when
 $Q \neq \text{NOT } \bar{Q}$

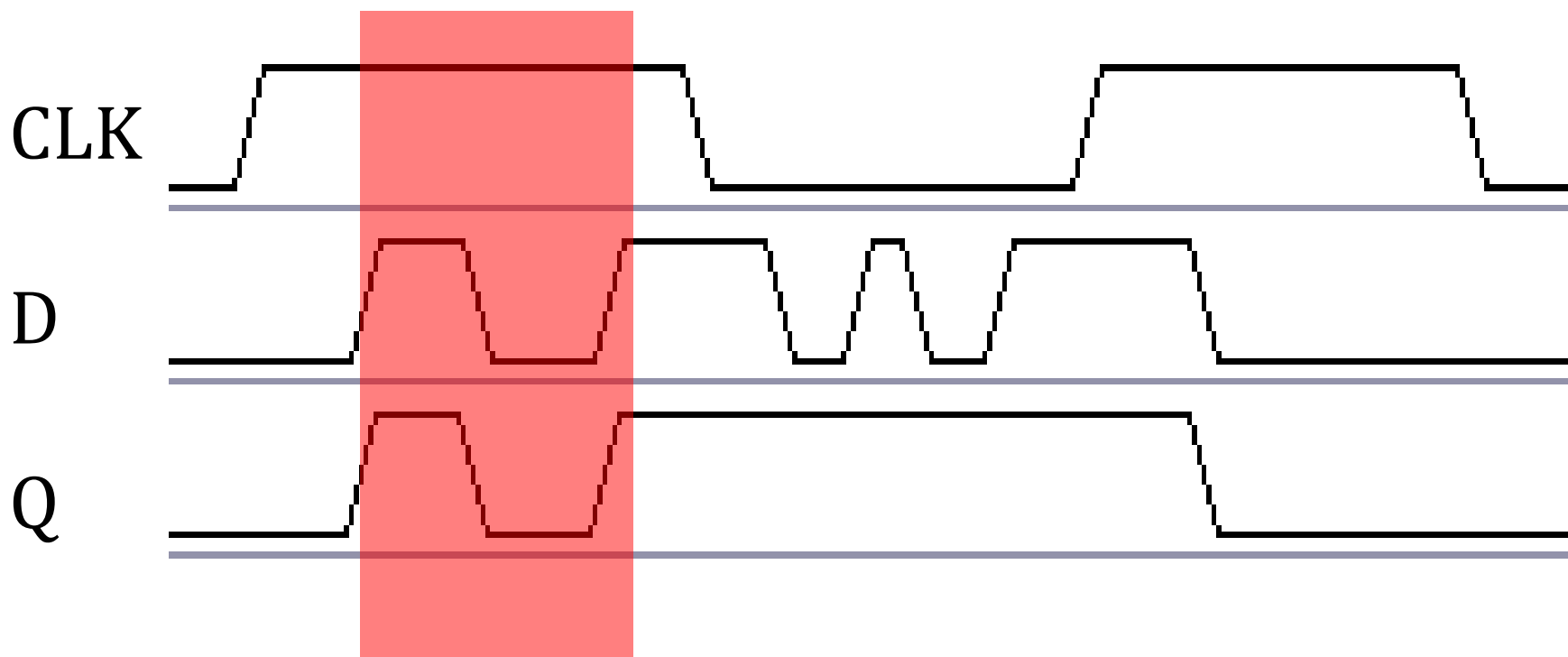
D Latch
Symbol



CLK	D	Q
0	X	Q_{prev}
1	D	D

D Latch的局限

- 本质是电平缓冲器
 - ▣ 在一个时钟周期内，Q可以随D多次翻转
- CLK还没有完全达到对**时钟**的设计初衷



D Flip-Flop Internal Circuit

- Two back-to-back latches (L1 and L2) controlled by complementary clocks

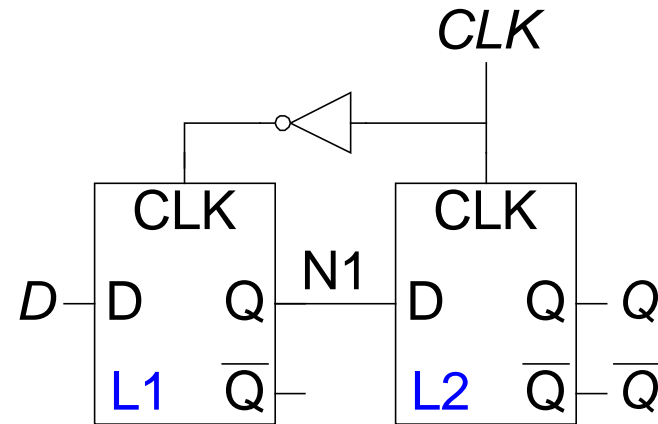
- When $CLK = 0$

- L1 is transparent
- L2 is opaque
- D passes through to N1

- When $CLK = 1$

- L2 is transparent
- L1 is opaque
- N1 passes through to Q

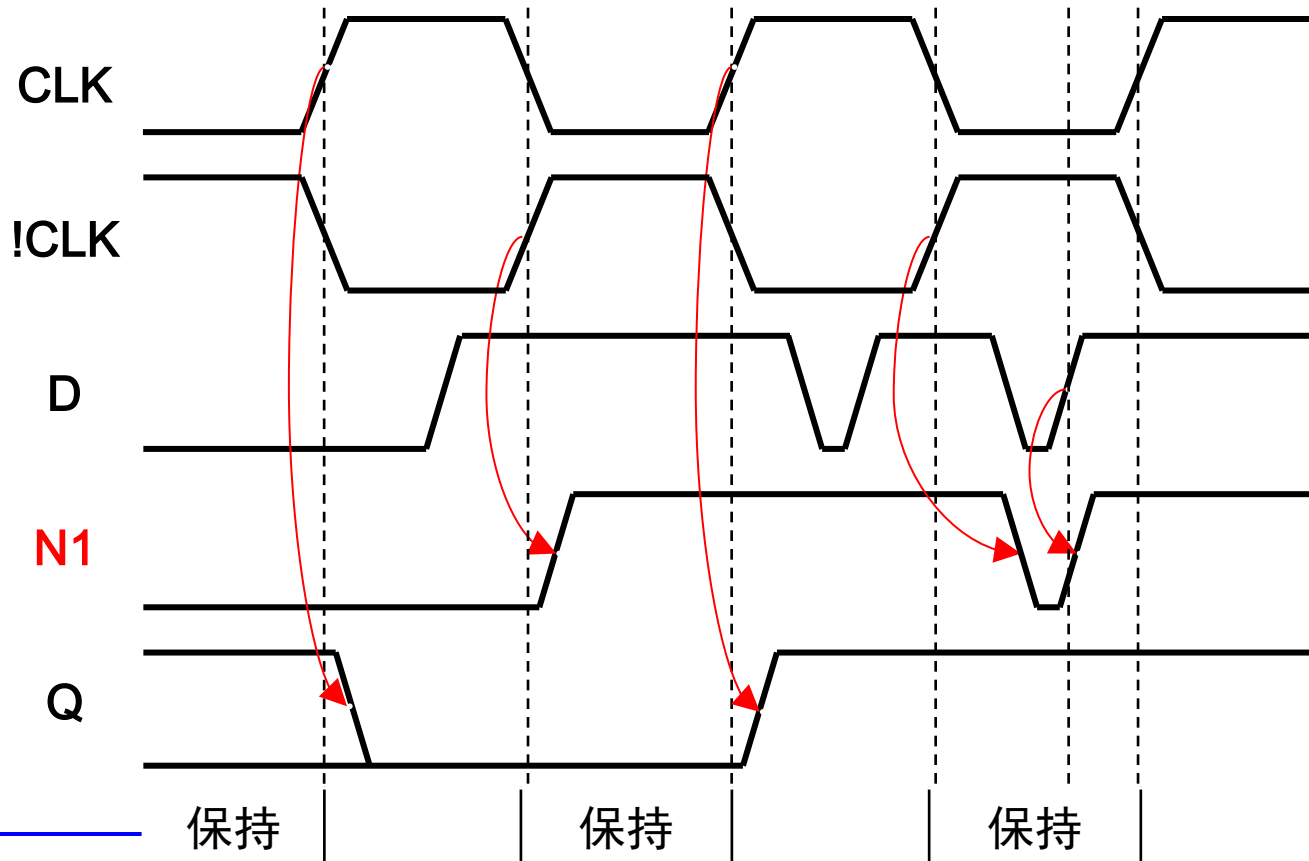
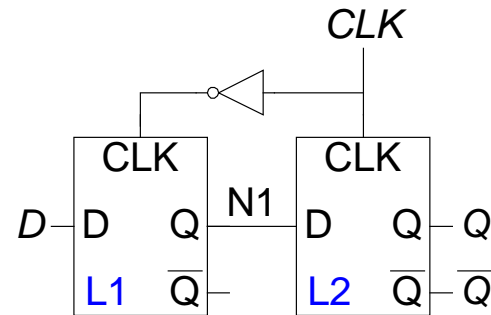
- Thus, on the edge of the clock (when CLK rises from $0 \rightarrow 1$)
 - D passes through to Q



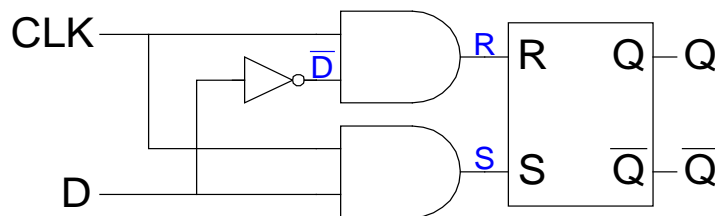
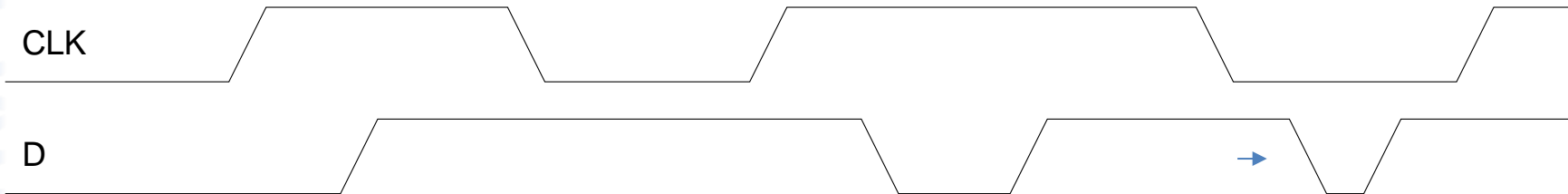
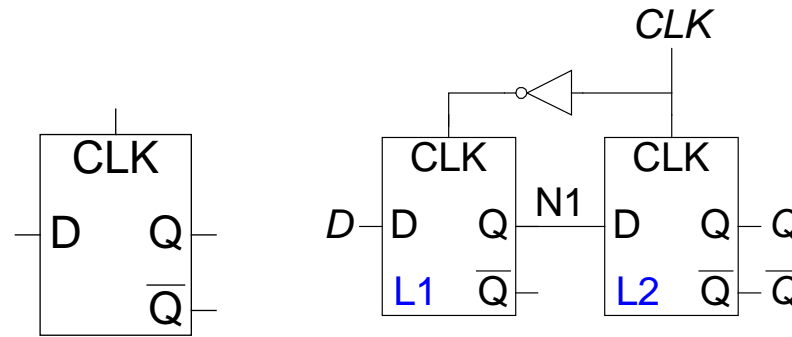
CLK	D	\overline{D}	S	R	Q	\overline{Q}
0	X	\overline{X}	0	0	Q_{prev}	\overline{Q}_{prev}
1	0	1	0	1	0	1
1	1	0	1	0	1	0

D-FF

CLK	D	\overline{D}	S	R	Q	\overline{Q}
0	X	\overline{X}	0	0	Q_{prev}	\overline{Q}_{prev}
1	0	1	0	1	0	1
1	1	0	1	0	1	0

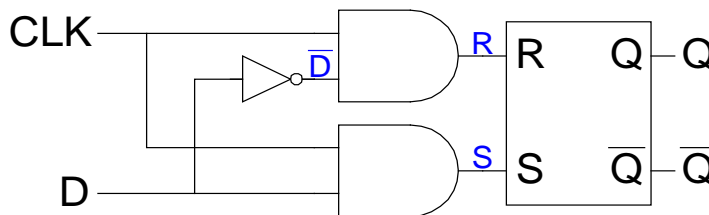
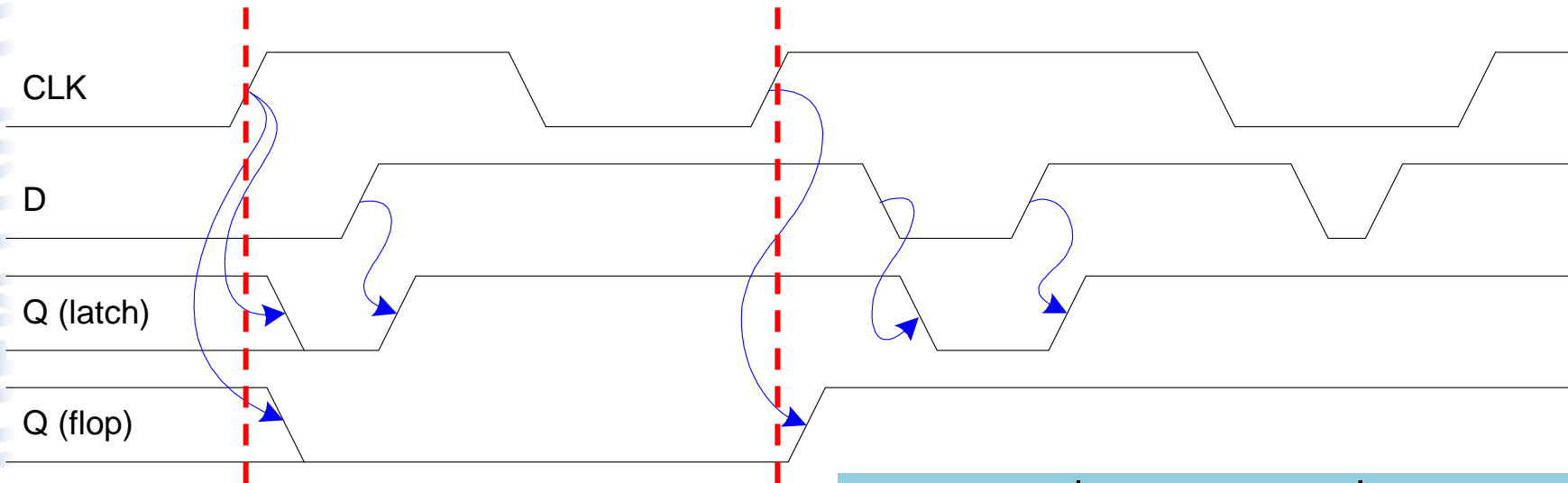
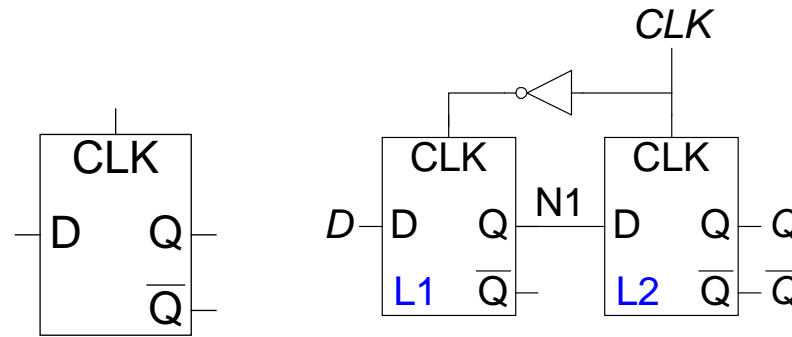


D Latch vs. D Flip-Flop



CLK	D	\overline{D}	S	R	Q	\overline{Q}
0	X	\overline{X}	0	0	Q_{prev}	\overline{Q}_{prev}
1	0	1	0	1	0	1
1	1	0	1	0	1	0

D Latch vs. D Flip-Flop

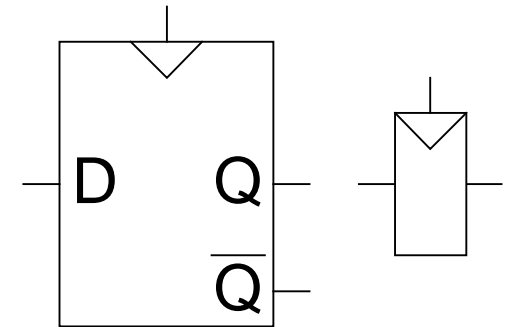


CLK	D	\overline{D}	S	R	Q	\overline{Q}
0	X	\overline{X}	0	0	Q_{prev}	\overline{Q}_{prev}
1	0	1	0	1	0	1
1	1	0	1	0	1	0


D Flip-Flop

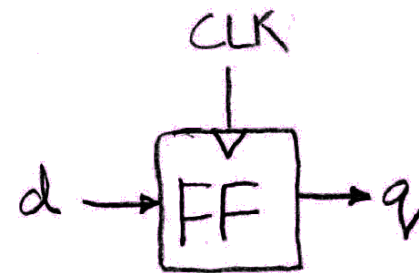
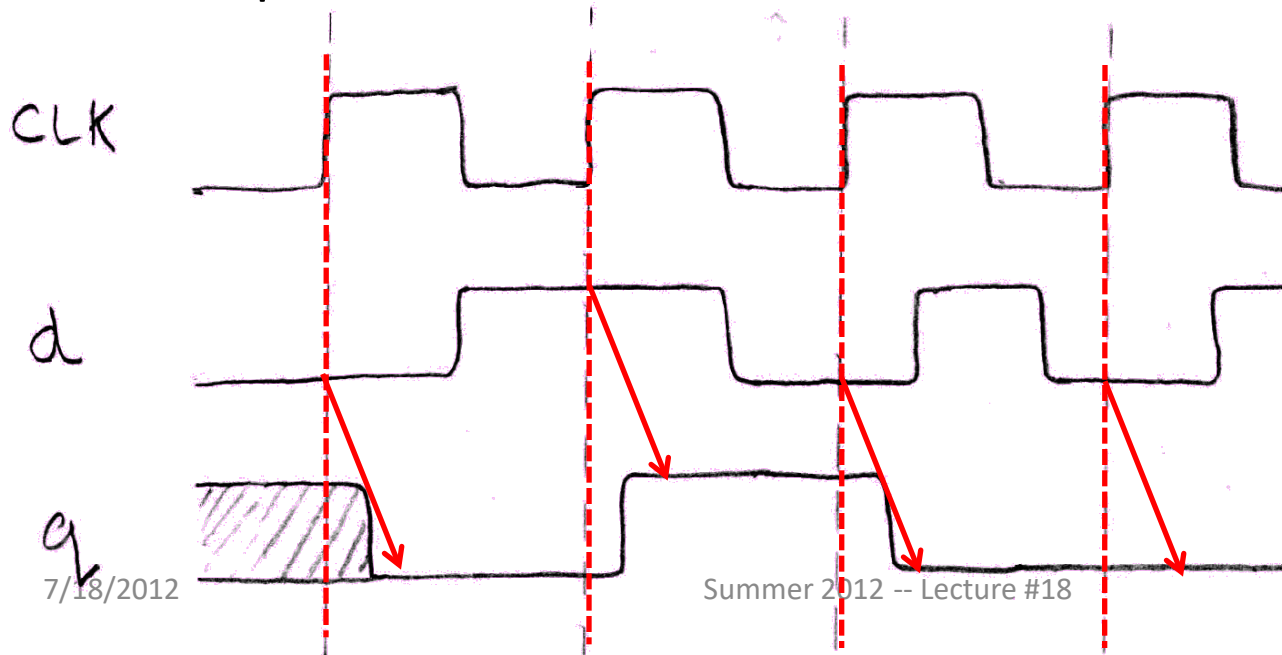
- **Inputs:** CLK , D
- **Function**
 - Samples D on rising edge of CLK
 - When CLK rises from 0 to 1, D passes through to Q
 - Otherwise, Q holds its previous value
 - Q changes only on rising edge of CLK
- Called *edge-triggered*
- Activated on the clock edge

D Flip-Flop Symbols



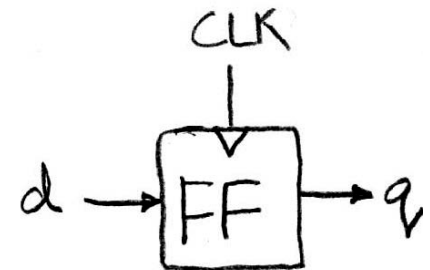
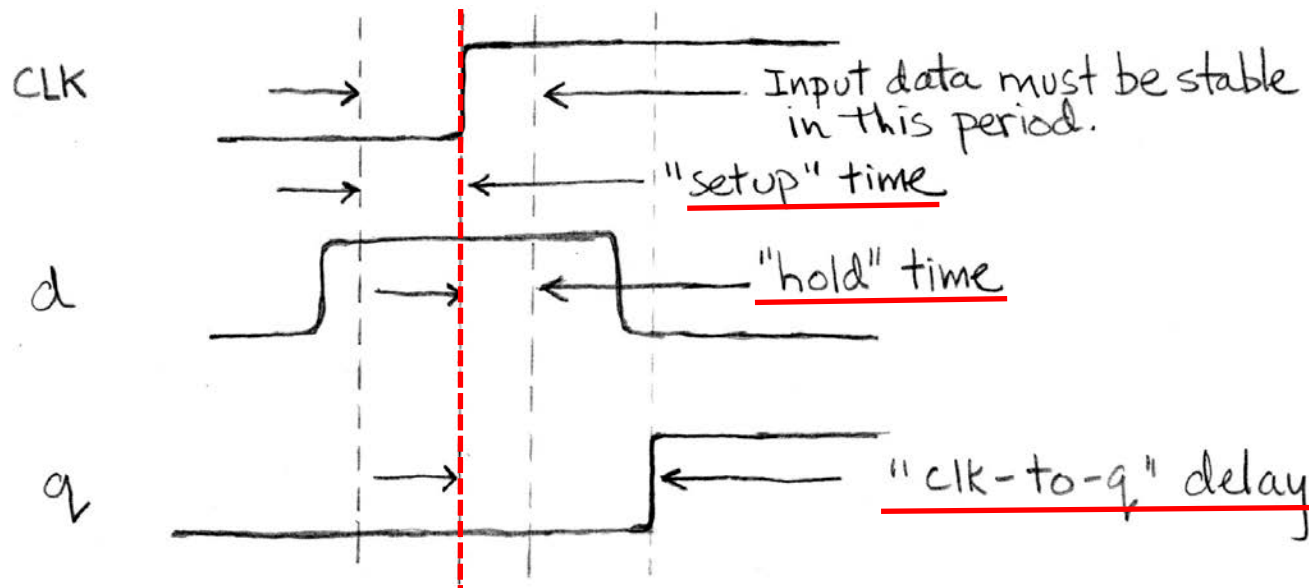
Flip-Flop Timing Behavior (1/2)

- Edge-triggered D-type flip-flop
 - This one is “positive edge-triggered” 
- “On the rising edge of the clock, input d is sampled and transferred to the output. At other times, the input d is ignored and the previously sampled value is retained.”
- Example waveforms:



Flip-Flop Timing Behavior (2/2)

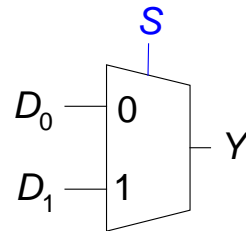
- **Setup Time:** how long the input must be stable *before* the CLK trigger for proper input read
- **Hold Time:** how long the input must be stable *after* the CLK trigger for proper input read
- **“CLK-to-Q” Delay:** how long it takes the output to change, measured from the CLK trigger



Multiplexer (Mux)

- Selects between one of N inputs to connect to output
- $\log_2 N$ -bit select input – control input
- Example:

2:1 Mux



S	D_1	D_0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

S	Y
0	D_0
1	D_1

压缩表

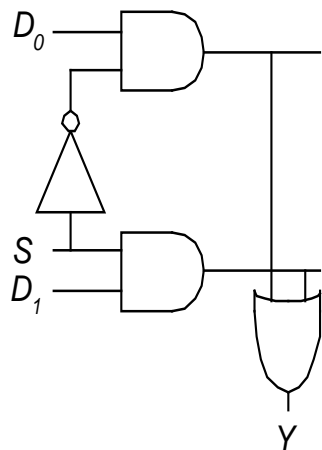
Multiplexer Implementations

- Logic gates**

- Sum-of-products form

		$D_0 D_1$			
S	Y	00	01	11	10
	0	0	0	1	1
	1	0	1	1	0

$$Y = D_0 \bar{S} + D_1 S$$



S	D_1	D_0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Q:

4选1 MUX逻辑表达式?

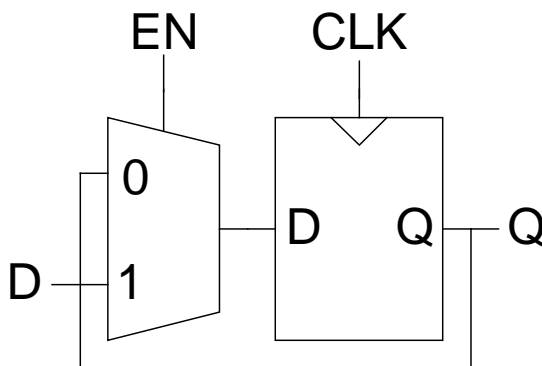
A:

- 1) 4选1, 故需要2个控制信号
- 2) 输入为2控制信号的压缩表

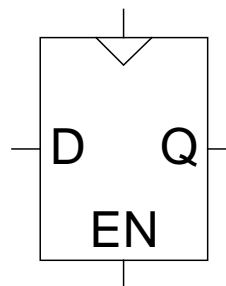
Enabled Flip-Flops

- **Inputs:** CLK , D , EN
 - The enable input (EN) controls when new data (D) is stored
- **Function**
 - $EN = 1$: D passes through to Q on the clock edge
 - $EN = 0$: the flip-flop retains its previous state

Internal
Circuit



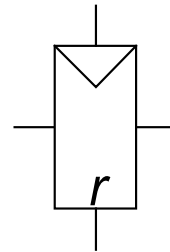
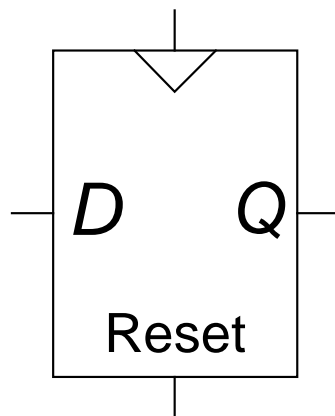
Symbol



Resettable Flip-Flops

- **Inputs:** CLK , D , $Reset$
- **Function:**
 - $Reset = 1$: Q is forced to 0
 - $Reset = 0$: flip-flop behaves as ordinary D flip-flop

Symbols

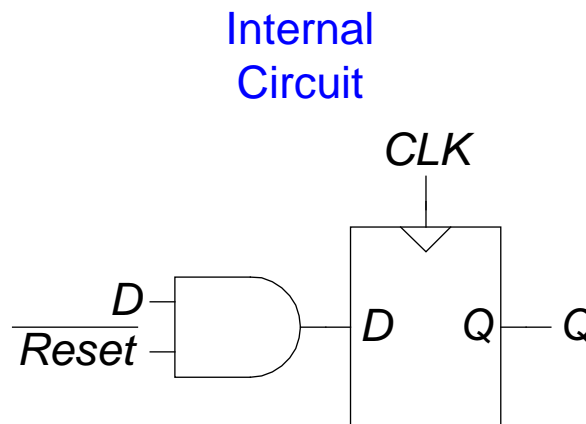


Resettable Flip-Flops

- Two types:
 - **Synchronous:** resets at the clock edge only
 - **Asynchronous:** resets immediately when $Reset = 1$
- Asynchronously resettable flip-flop requires changing the internal circuitry of the flip-flop
- Synchronously resettable flip-flop?

Resettable Flip-Flops

- Two types:
 - **Synchronous:** resets at the clock edge only
 - **Asynchronous:** resets immediately when $Reset = 1$
- Asynchronously resettable flip-flop requires changing the internal circuitry of the flip-flop
- Synchronously resettable flip-flop



作业

- 学习在Logisim中输入真值表，自动综合出组合逻辑。电路功能如下：
 - 输入S[2:0]与输出Q[2:0]对应关系如下
 - ◆ 000: 010
 - ◆ 001: 101
 - ◆ 010: 110
 - ◆ 011: 111
 - ◆ 100: 000
 - ◆ 101: 001
 - ◆ 110: 010
 - ◆ 111: 011
 - 阅读User's Guide中的Combinational analysis，然后在Logisim中输入真值表，自动产生电路

作业

- 用组合逻辑设计一个可以完成循环左移功能的8位移位器
 - D[7:0]: 移位器的8位输入
 - Q[7:0]: 移位器的8位输出
 - S[2:0]: 循环移位控制
 - ◆ S[2:0]=000b, 无循环左移
 - ◆ S[2:0]=001b, 循环左移1位
 - ◆ 依次类推
 - ◆ S[2:0]=111b, 循环左移7位
 - WORD: 给出Q7~Q0的逻辑表达式
 - Logisim: 测试验证 (不提交)

作业

《数字设计》	Logisim (不提交)	WORD
2.24	✓	✓
3.1~3.3		✓
3.5	✓	✓

作业

■ 《数字设计与计算机体系结构》

《数字设计》	Logisim (不提交)	WORD
3.6	✓	1、给出真值表 2、简要说明工作原理
3.9	✓	
3.10	✓	